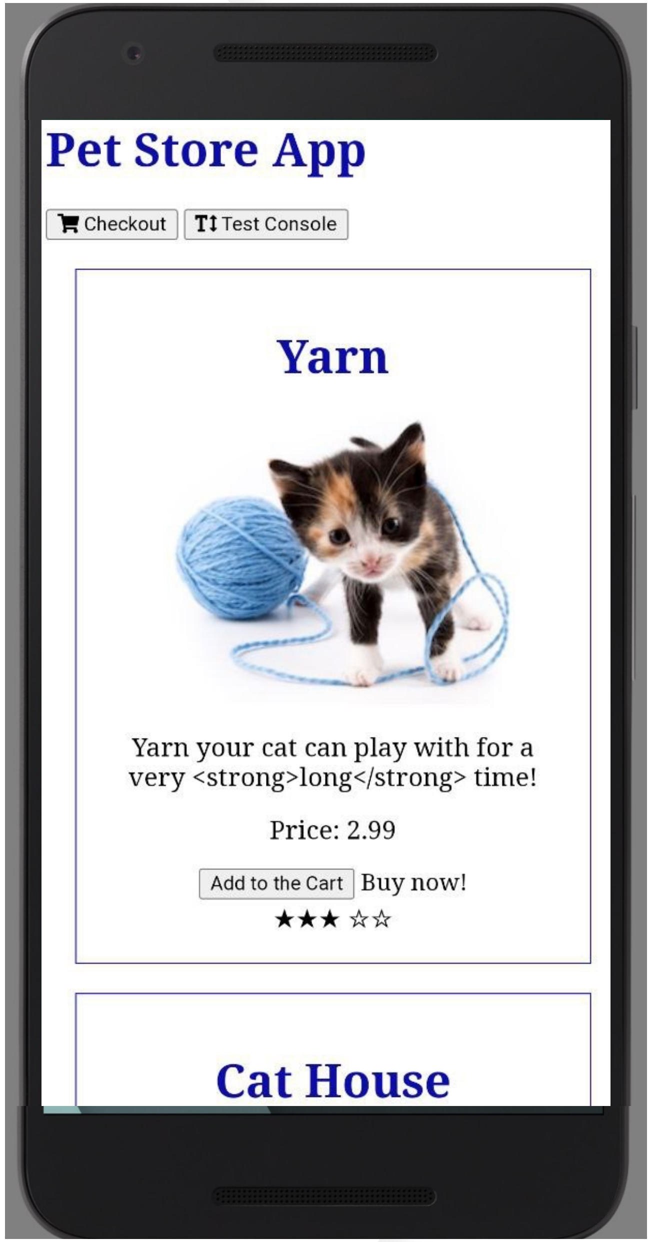
Progressive Web Apps (PWA)



Pet Store App

Yarn

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

Buy now!

★★★ ☆☆

Cat House

Outline and Learning Objectives

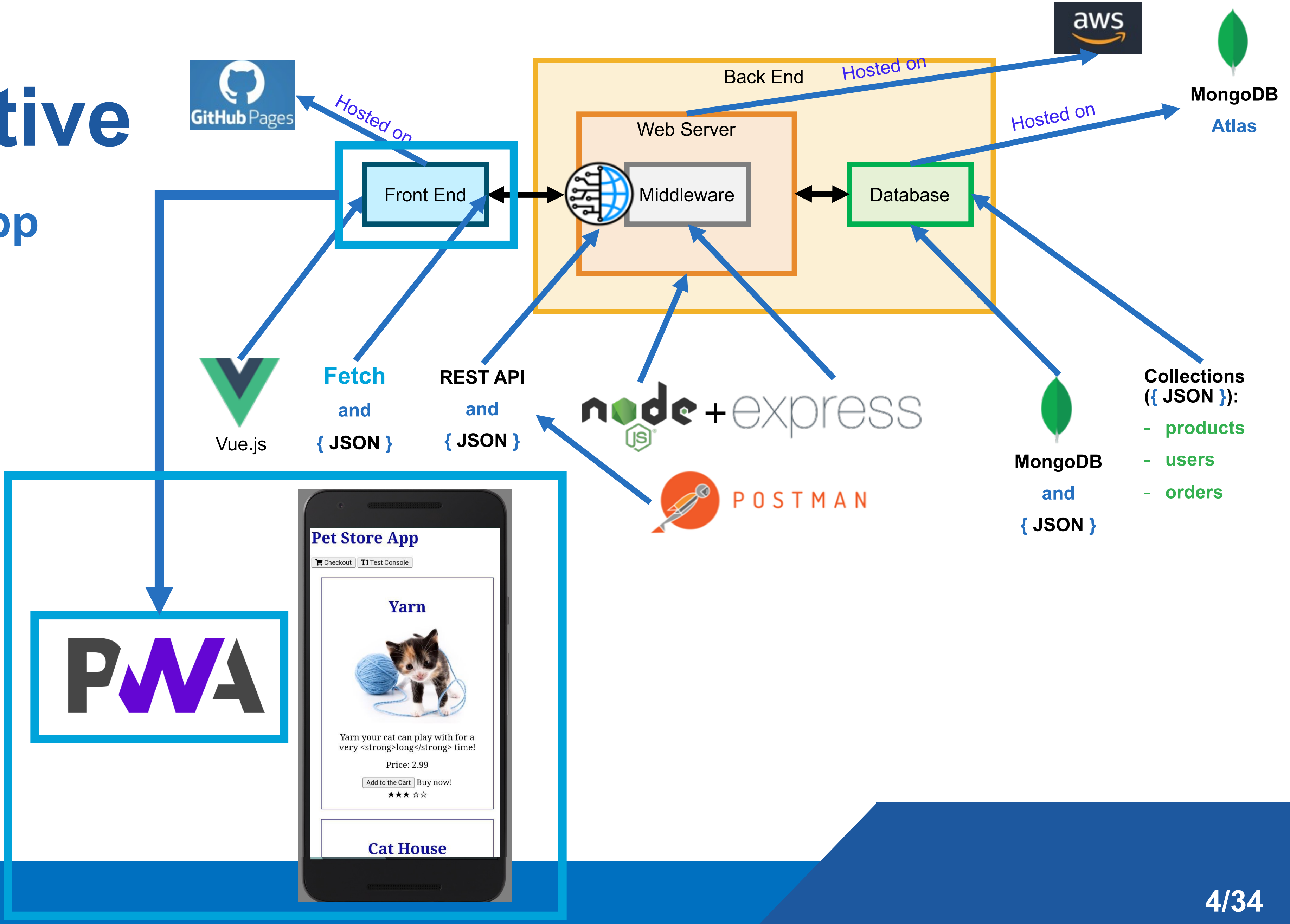
- **Progressive Web Apps (PWA):**
  - to understand the basics on PWA
  - to understand the basics on how to turn a Web App into a PWA
  - to understand the basics on creating and using Service Workers for PWA
  - to understand the basics on Static Caching with Service Workers and PWA
  - to understand the basics on Dynamic Caching with Service Workers and PWA
  - to understand the basics on how to test and use the PWA online and offiline, locally, on a computer and on a mobile device
  - to understand the basics on potential problems with PWA and HTTPS

- **Suggestions for Reading**

Progressive Web Apps (PWA): Intro

Our Objective

To turn our **Web App** into a **Mobile App**



GitHub Pages

Hosted on

Front End

Back End

Hosted on

aws

Web Server

Middleware

Database

Hosted on

MongoDB Atlas

Vue.js

Fetch and { JSON }

REST API and { JSON }

node JS + express

POSTMAN

MongoDB and { JSON }

Collections ({ JSON }):

- products
- users
- orders

PWA

Pet Store App

Checkout | Test Console

Yarn

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

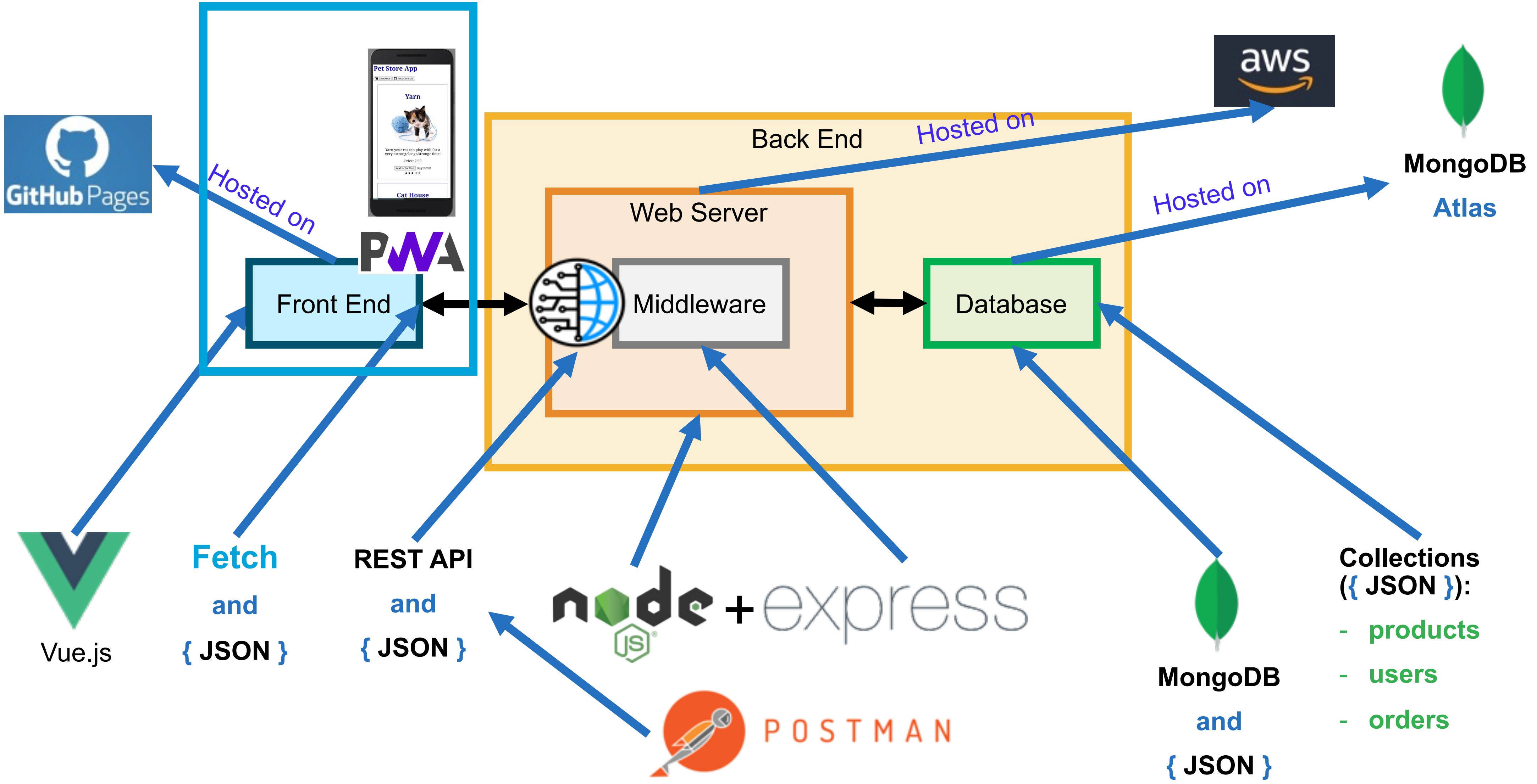
Add to the Cart Buy now!

★★★☆☆

Cat House

The Big Picture and PWA

We will cover in this section: PWA



PWA

Front End

Hosted on

GitHub Pages

Back End

Hosted on

aws

Web Server

Middleware

Database

Hosted on

MongoDB

Atlas

Vue.js

Fetch

and

{ JSON }

REST API

and

{ JSON }

node + express

POSTMAN

MongoDB

and

{ JSON }

Collections ({ JSON }):

- products
- users
- orders

What are Progressive Web Apps (PWA)?

- **Modern Web APIs** + **progressive enhancement** = **cross-platform apps**

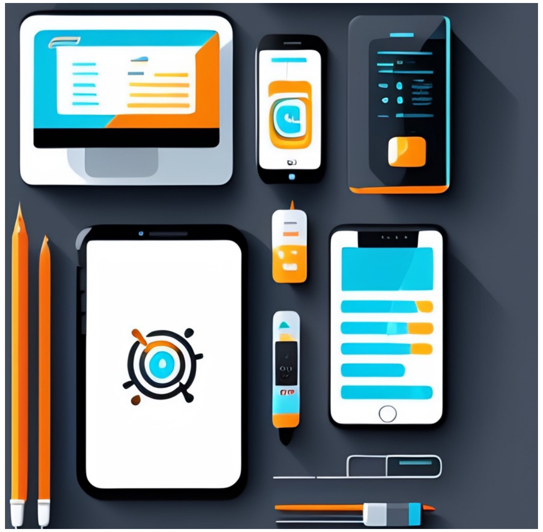
- **Work everywhere**: desktop, tablet, and phone

- (Almost) the **same user experience** and **advantages** as **native apps**

Have both web and native app features

- **As a web app**:
  - **easier and faster to visit a website** than installing an application;
  - you can **share web apps via a link**.

- **As a native app**:
  - can **install it locally**;
  - use **home screen icons** to access the app;
  - works **offline**.



Additional Features

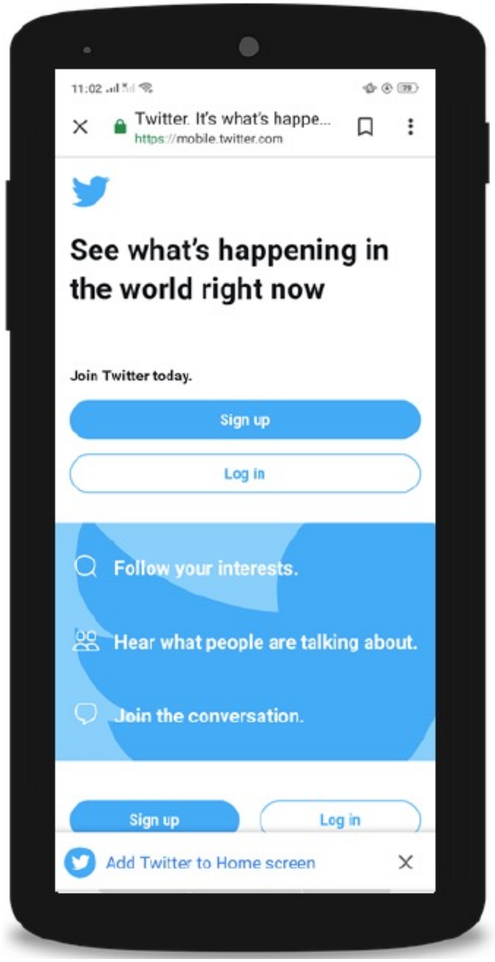
- **Discoverable**, can be found through **search engines**;
- **Progressive**, still **usable on older browsers**, and **fully-functional on the latest ones**;
- **Responsive**, **usable on any device** with a **browser** - phones, tablets, laptops, TVs, fridges, etc.;
- **Secure**, connection between your device and app server is secured against any third parties trying to get access to your sensitive data.

HTTPS

Examples: Twitter Lite

Twitter Lite

- **Reachable:** via **search engines**
- **Shareable:** via a **link**
- **Installable:** via the **web page**
- **Available:** via a **home screen app icon**
- **Runnable:** both **online** and **offline** (offline for all or **most of the aspects**, depending on the **specific App**)



11:02

Twitter. It's what's happe...
https://mobile.twitter.com

See what's happening in the world right now

Join Twitter today.

Sign up

Log in

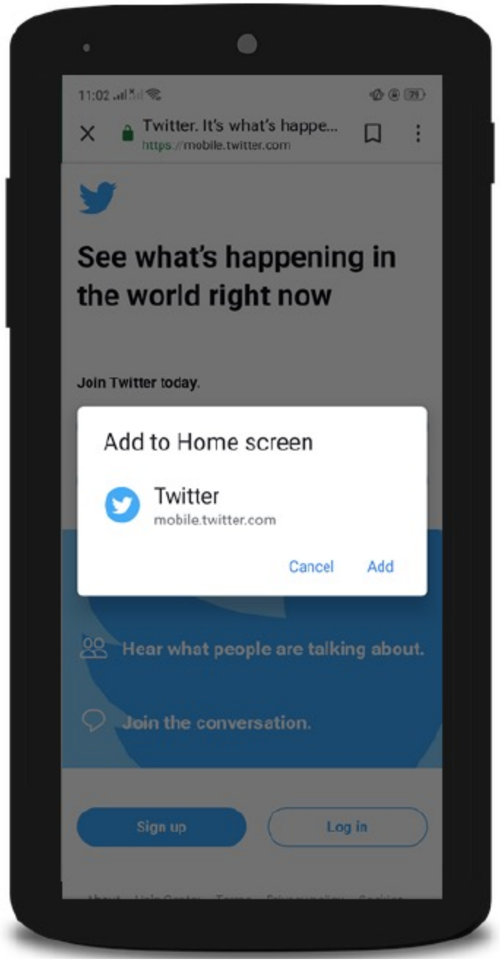
Follow your interests.

Hear what people are talking about.

Join the conversation.

Sign up   Log in

Add Twitter to Home screen



11:02

Twitter. It's what's happe...
https://mobile.twitter.com

See what's happening in the world right now

Join Twitter today.

Add to Home screen

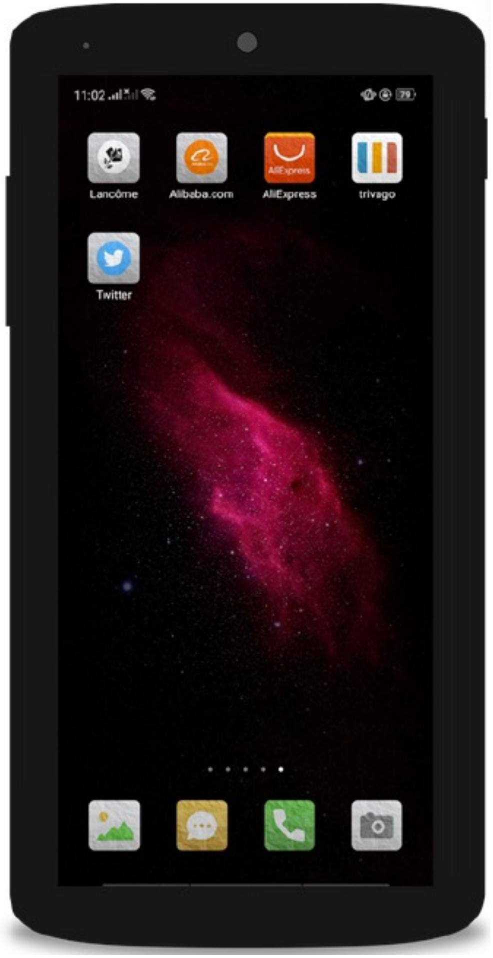
Twitter
mobile.twitter.com

Cancel   Add

Hear what people are talking about.

Join the conversation.

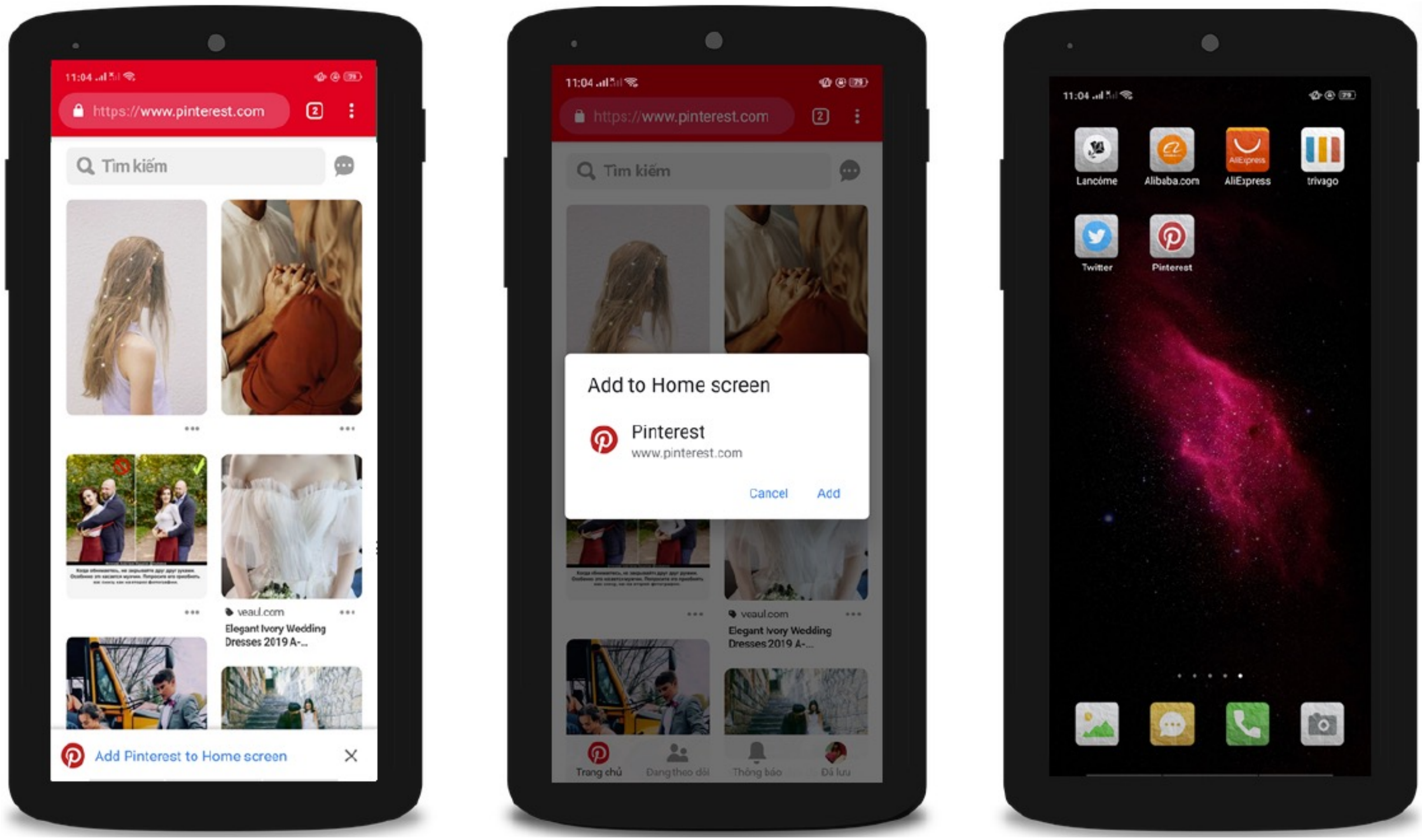
Sign up   Log in



Examples: Pinterest

Pinterest

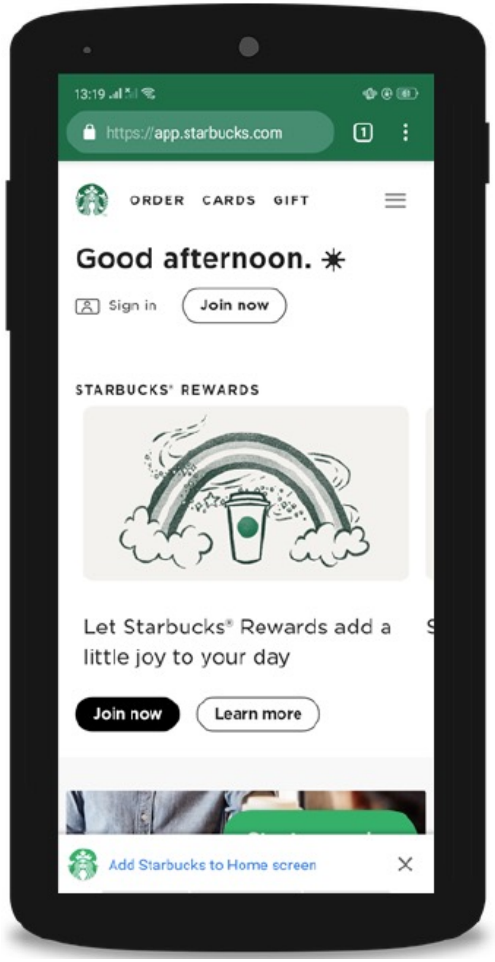
- **Reachable:** via **search engines**
- **Shareable:** via a **link**
- **Installable:** via the **web page**
- **Available:** via a **home screen app icon**
- **Runnable:** both **online** and **offline** (offline for all or **most of the aspects**, depending on the **specific App**)



Examples: Starbucks

Starbucks

- **Reachable:** via **search engines**
- **Shareable:** via a **link**
- **Installable:** via the **web page**
- **Available:** via a **home screen app icon**
- **Runnable:** both **online** and **offline** (offline for all or **most of the aspects**, depending on the **specific App**)

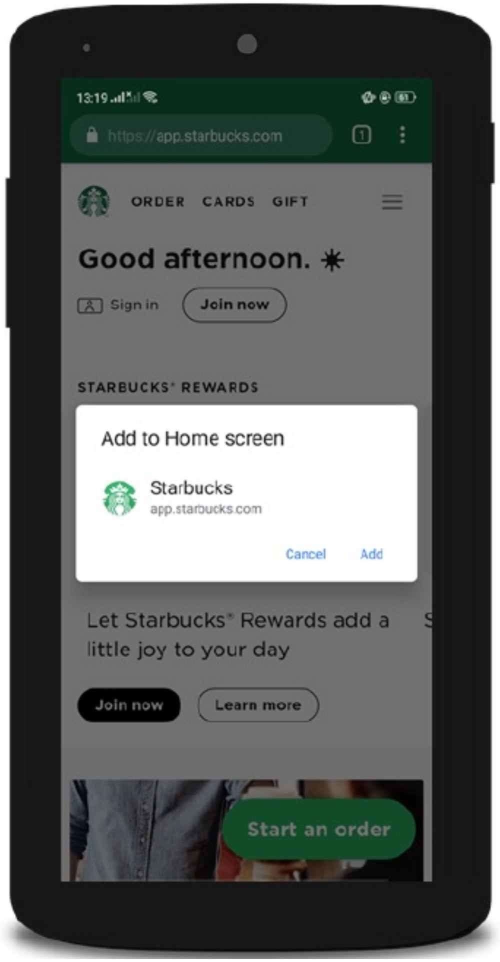


https://app.starbucks.com

Good afternoon. ☀

Let Starbucks® Rewards add a little joy to your day

Add Starbucks to Home screen



https://app.starbucks.com

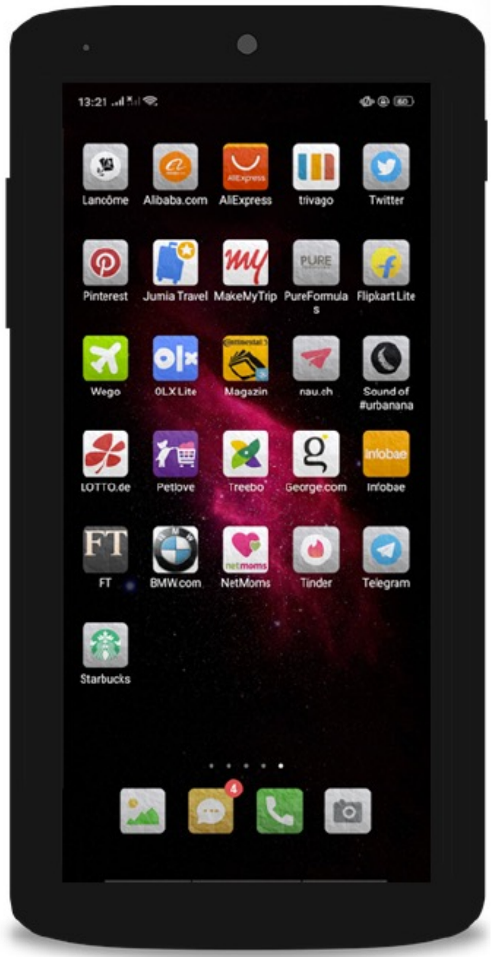
Good afternoon. ☀

Add to Home screen

Starbucks
app.starbucks.com

Cancel Add

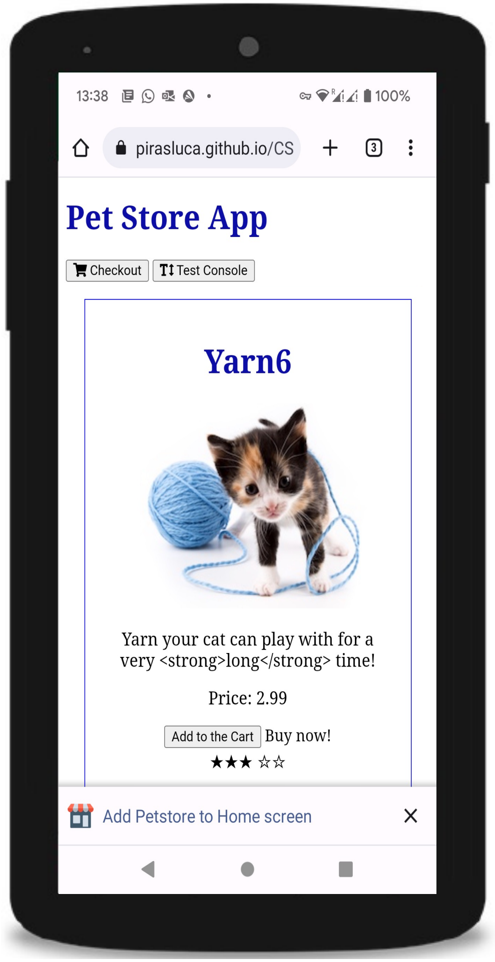
Let Starbucks® Rewards add a little joy to your day



Examples: Our App

Our App

- **Reachable:** via **search engines**
- **Shareable:** via a **link**
- **Installable:** via the **web page**
- **Available:** via a **home screen app icon**
- **Runnable:** both **online** and **offline** (offline for all or **most of the aspects**, depending on the **specific App**)



Pet Store App

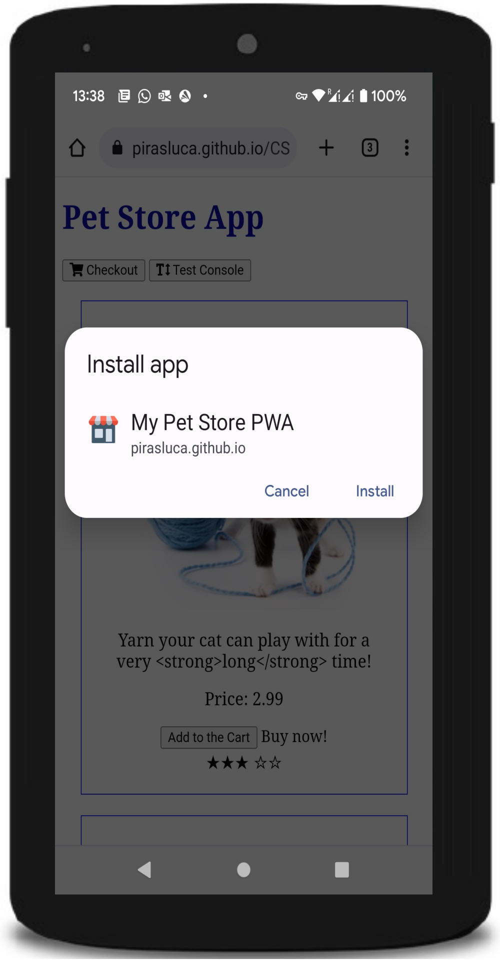
Yarn6

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

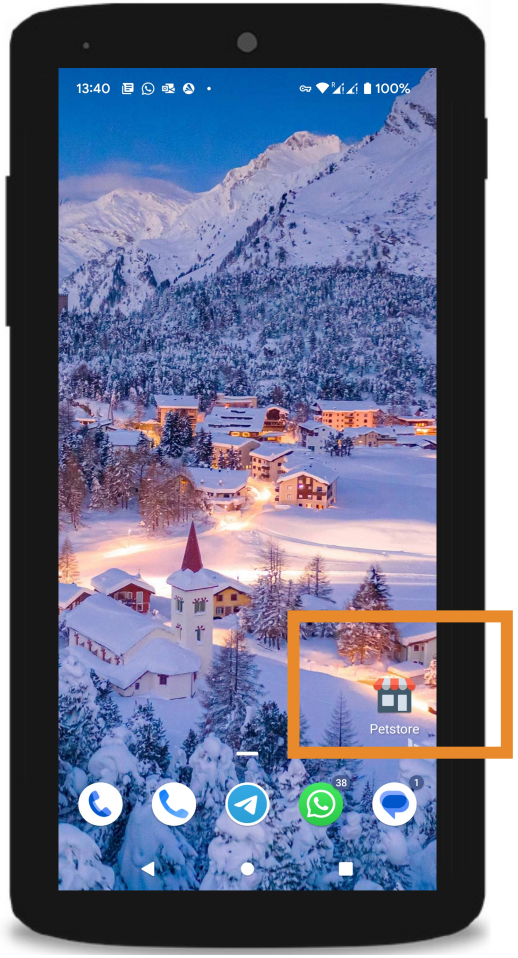
Add to the Cart Buy now!

★★★☆☆



Install app

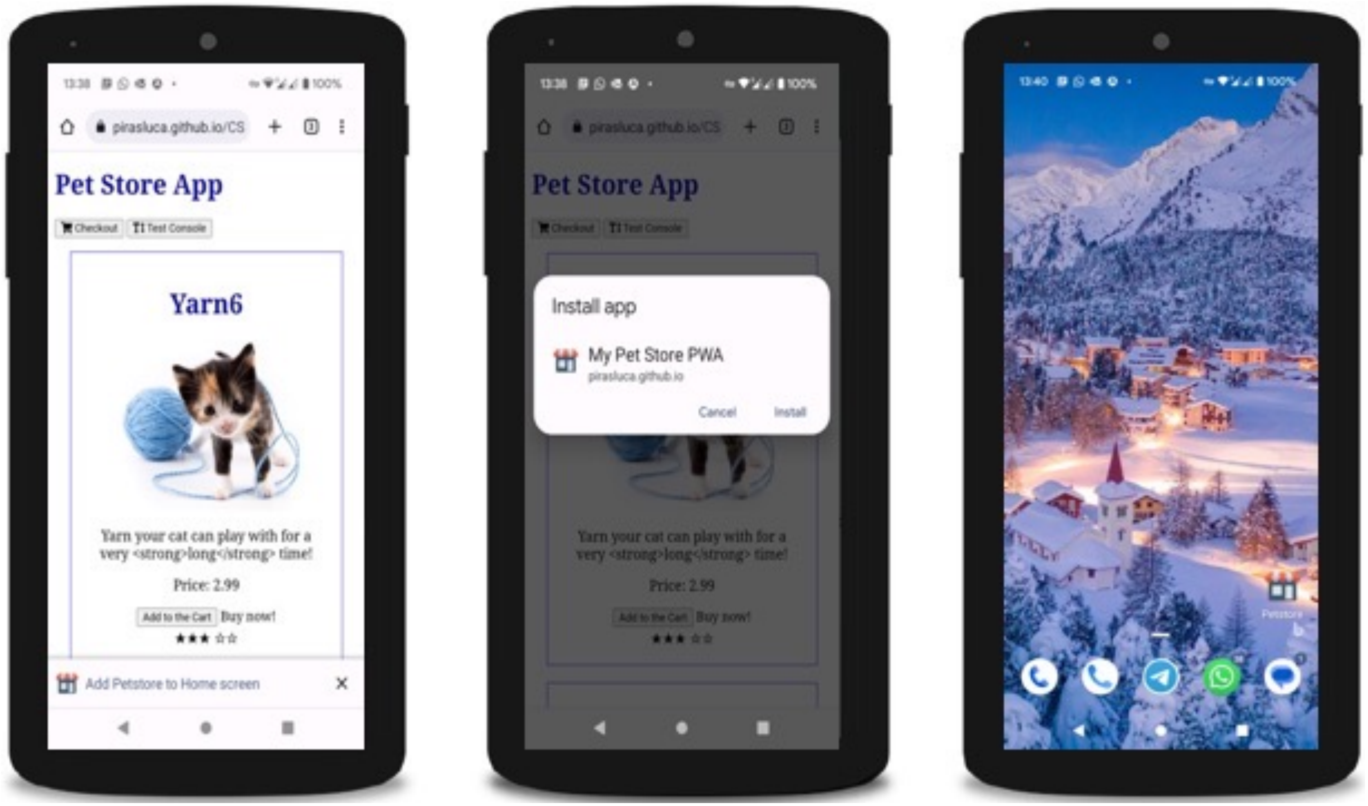
My Pet Store PWA
pirasluca.github.io



Our Approach

Our Approach

- We will continue with the **“Pet Store” example from CW1**:
    - start with **loading the data locally from a .js file** (**then** also **remotely from the Back End**)
- Turn it into a **PWA**:
    - can install locally
    - add a shortcut to the home screen
    - work offline
- We can **test on computers**, **emulators**, **phones**, but also on other devices:
    - The **demonstration** will be on **computers** and **emulators** (not easy to share the phone screen)



pirasluca.github.io/CS

Pet Store App

Checkout | Test Console

Yarn6

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

Add to the Cart | Buy now!

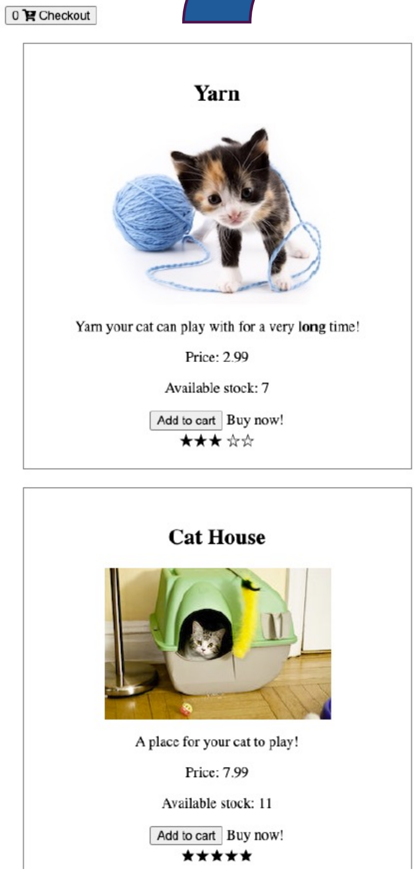
★★★☆☆

Add Petstore to Home screen

Install app

My Pet Store PWA
pirasluca.github.io

Cancel | Install



0 🛒 Checkout

Yarn

Yarn your cat can play with for a very **long** time!

Price: 2.99

Available stock: 7

Add to cart Buy now!
★★★☆☆

Cat House

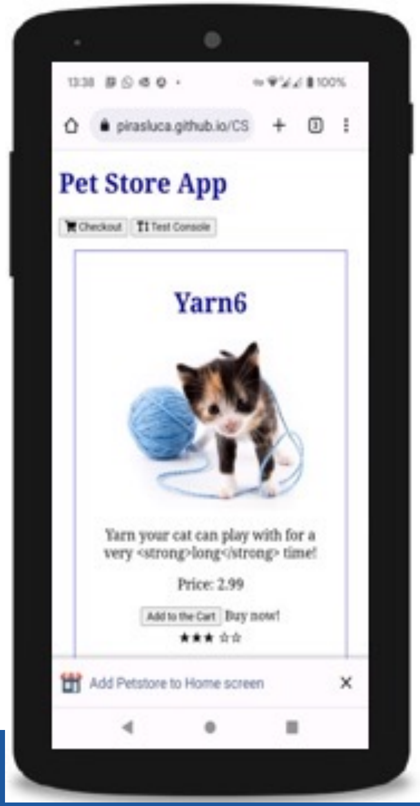
A place for your cat to play!

Price: 7.99

Available stock: 11

Add to cart Buy now!
★★★★★

From Web App
to
Progressive Web App (PWA)



13:38 100%

pirasluca.github.io/CS

Pet Store App

🛒 Checkout ⇅ Test Console

Yarn6

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

Add to the Cart Buy now!
★★★☆☆

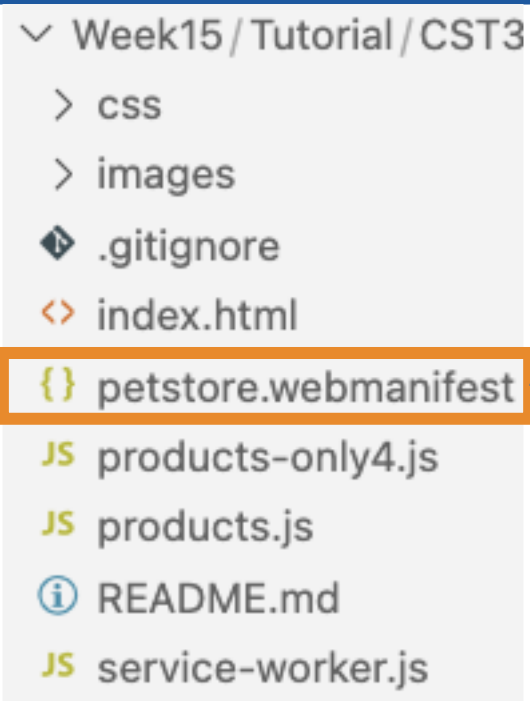
Add Petstore to Home screen ✕

How to make PWAs Installable?

Requirements for an Installable PWA:

- **[Web Manifest]** a web manifest file, with the correct fields
- **[HTTPS]** the website needs to be on a **server with HTTPS connection**
  - we will use **GitHub Pages**, which provide HTTPS connection
- **[App Icon]** an icon (image file) to represent the app on the device
- **[Service Worker]** a registered service worker, to make the app work offline
  - requires some coding

PWA: The Web Manifest File



Week15 / Tutorial / CST3

- css
- images
- .gitignore
- index.html
- petstore.webmanifest
- products-only4.js
- products.js
- README.md
- service-worker.js

The Web Manifest File:

```
<link rel="manifest" href="petstore.webmanifest">
```

- needs to be **linked in the <head> section** of the html file
- lists all the **information** about the PWA in a **JSON** format;
- usually resides in the **root folder** of a web app;
- contains **useful information**, such as
  - the app's name,
  - paths to different-sized icons (for example, as the home screen icon),
  - a background colour to use in loading or splash screens;
- has an extension of **.webmanifest**;

```
{
    "name": "My Pet Store PWA",
    "short_name": "Petstore",
    "description": "My online pet store",
    "icons": [
        {
            "src": "images/icon-32.png",
            "sizes": "32x32",
            "type": "image/png"
        },
        {
            "src": "images/icon-512.png",
            "sizes": "512x512",
            "type": "image/png"
        }
    ],
    "start_url": "index.html",
    "display": "fullscreen",
    "background_color": "#260bca"
}
```

The App in the Browser

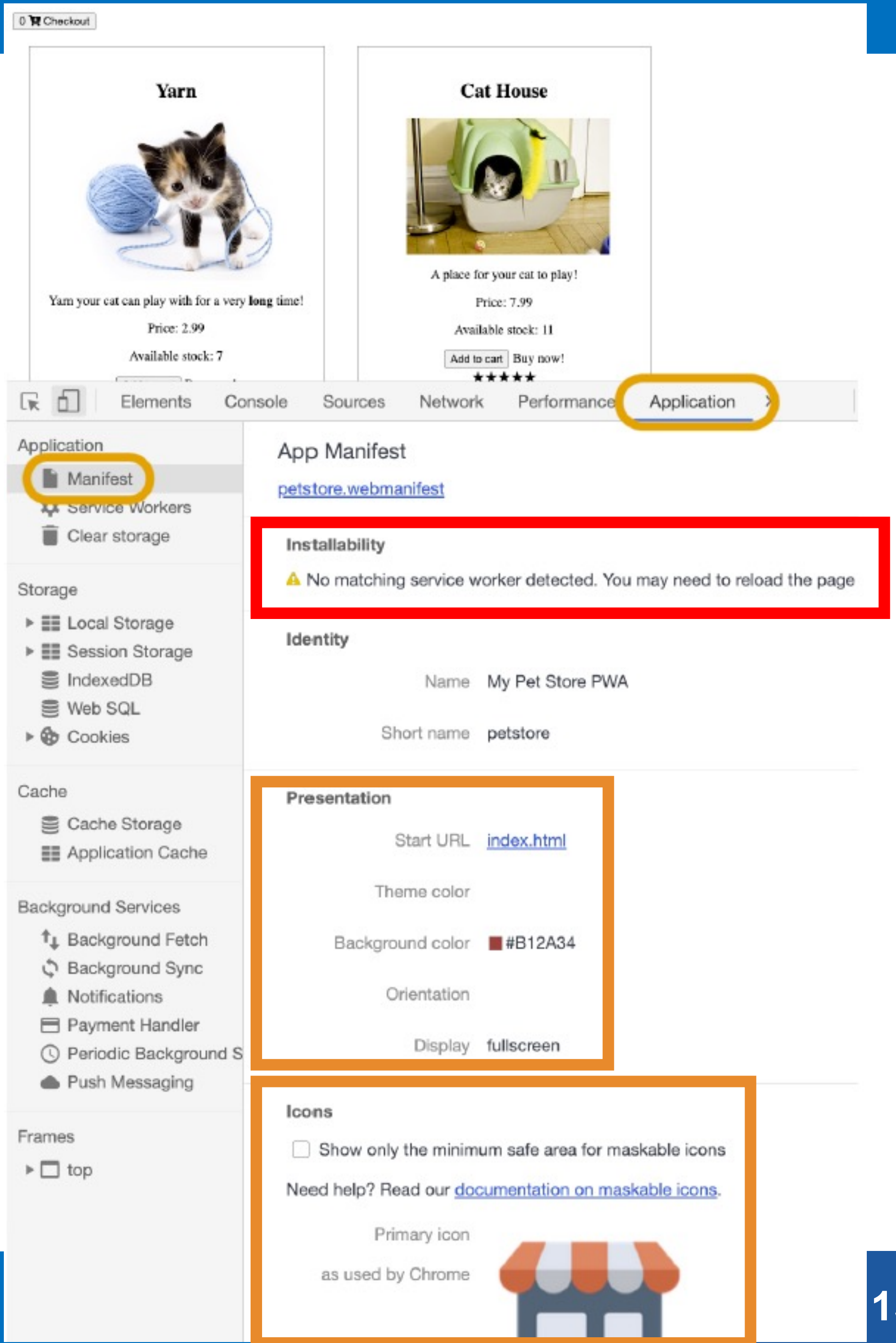
Let’s check the app in the browser, now that it has:

- **a web manifest file** and it **linked in the head**

- It seems nothing has changed if you open the app in the browser now.
- If you go to the browser **devtools -> “Application” -> “Manifest”**:
  - you can see **Chrome** recognised the **pestore.webmanifest file**
  - showing values such as **“name”**, **“start URL”**, and **“background colour”**

- However, there is still **an error “Installability”**: **No matching service worker detected'.**

Next Slide



Yarn

Yarn your cat can play with for a very **long** time!

Price: 2.99

Available stock: 7

Cat House

A place for your cat to play!

Price: 7.99

Available stock: 11

App Manifest

petstore.webmanifest

Installability

⚠ No matching service worker detected. You may need to reload the page

Identity

Name My Pet Store PWA

Short name petstore

Presentation

Start URL index.html

Theme color

Background color 🟥 #B12A34

Orientation

Display fullscreen

Icons

□ Show only the minimum safe area for maskable icons

Need help? Read our documentation on maskable icons.

Primary icon as used by Chrome

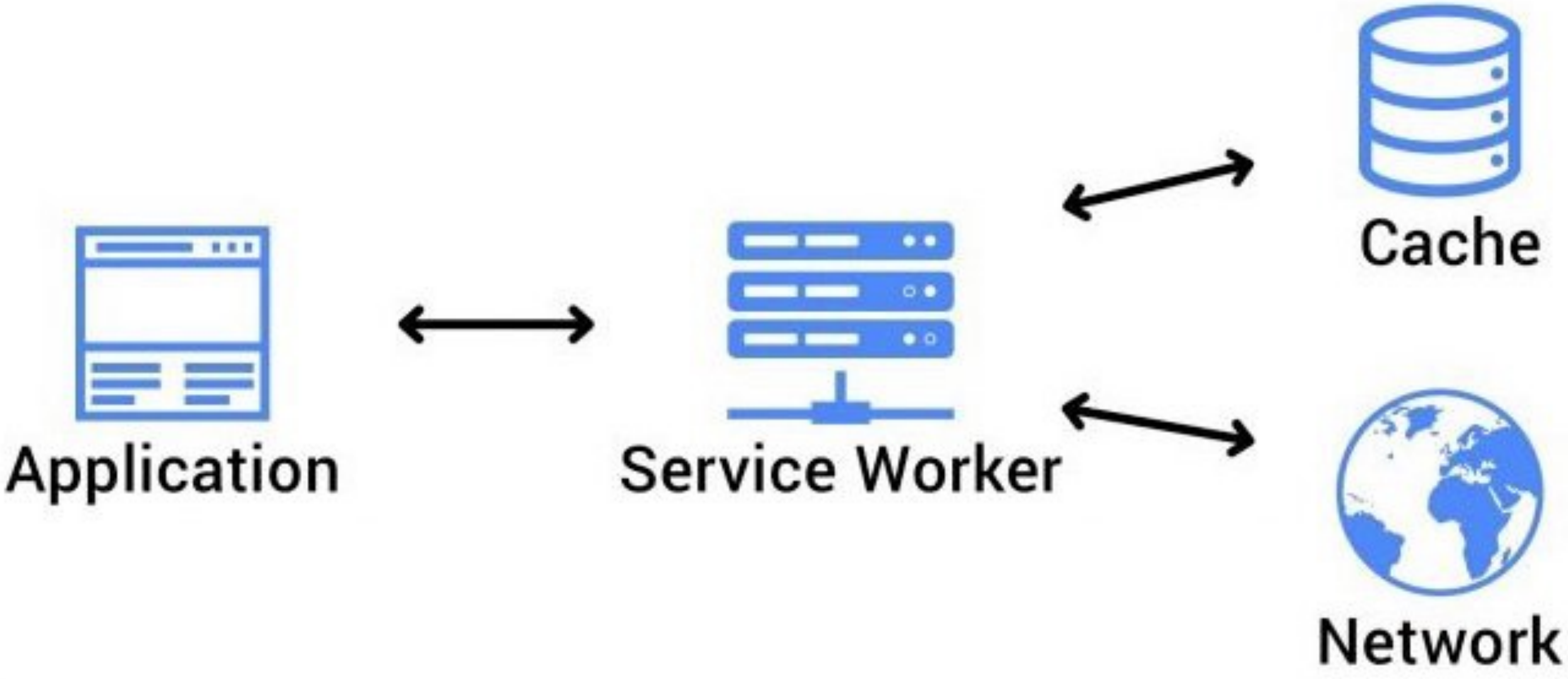
PWA, Service Workers and Static Caching

Service Workers

Service Workers (SW) are a **virtual proxy** between the browser and the network:

- SW can **cache the files of a website** and make them **available when the device is offline**;
- they **run on a separate thread** from the main JavaScript code;
    - do **not have any access to the DOM** structure;
- the **API is non-blocking**;
    - you can **give a Service Worker something to work on**, and receive the result when it is ready using a **Promise-based approach**;
- service Workers can **only be executed in secure contexts**.

HTTPS



Registering a Service Worker

Registering a Service Worker (in the main html file)

- If the **service worker API is supported in the browser**, it is registered using the `navigator.serviceWorker.register()` method.

- **Its content reside in the** `service-worker.js` file, and **can be executed after the registration is successful**.
- When **registration is complete**, the `service-worker.js` **file is automatically downloaded, then installed, and finally activated**.

- We can put this code in the **“created” section** of our Vue.js instance, or within a `<script>` **tag in the web page**

```
created: function() {
   if ("serviceWorker" in navigator) {
      navigator.serviceWorker.register("service-worker.js");
   }
...
```

Caching the Files

Caching the files (in the `service-worker.js` file)

- First, a variable for storing the cache name is created
- then, the files to be cached are listed in an array
- Then we can cache the files in the 'install' event listener
- The `self` here refers to the `window` object in JavaScript (the browser window).
- The service worker does not install until the code inside `waitUntil` is executed (returns a promise)
- The `caches` is a special 'Cache Storage' object for Service Worker to save data
- Here, we open a cache with a given name (`petstore-v1`),
- Then add all the files (listed in `cacheFiles`) to the cache, so they are available next time it loads.

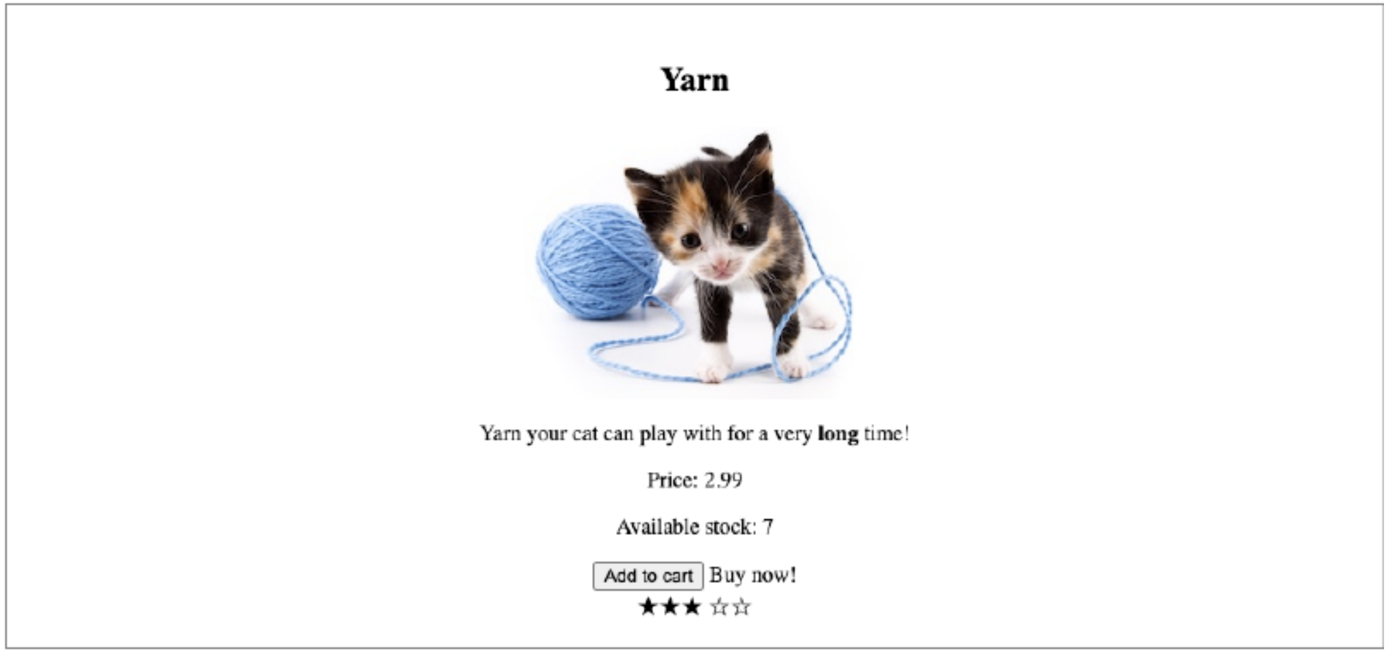
```
var cacheName = "petstore-v1";
var cacheFiles = [
   "index.html",
   //"products.js",
   "images/cat-house.jpg",
   "images/cat-litter.jpg",
   "images/icon-32.png",
   "images/icon-512.png",
   "images/laser-pointer.jpg",
   "images/yarn.jpg"
];

self.addEventListener("install", function(e) {
   console.log("[Service Worker] Install");
   e.waitUntil(
      caches.open(cacheName).then(function(cache) {
         console.log("[Service Worker] Caching files");
         return cache.addAll(cacheFiles);
      })
   );
});
```

Testing the Service Worker

Let’s test the ServiceWorker in the browser:

- If you open the app again in the browser, **the page looks the same**
- If you go to **devtools -> Application -> Service Workers**
- It should show that `serviceworker.js` is **activated** and **running**
- If there is any error, make sure the file names in the list are correct.



Yarn

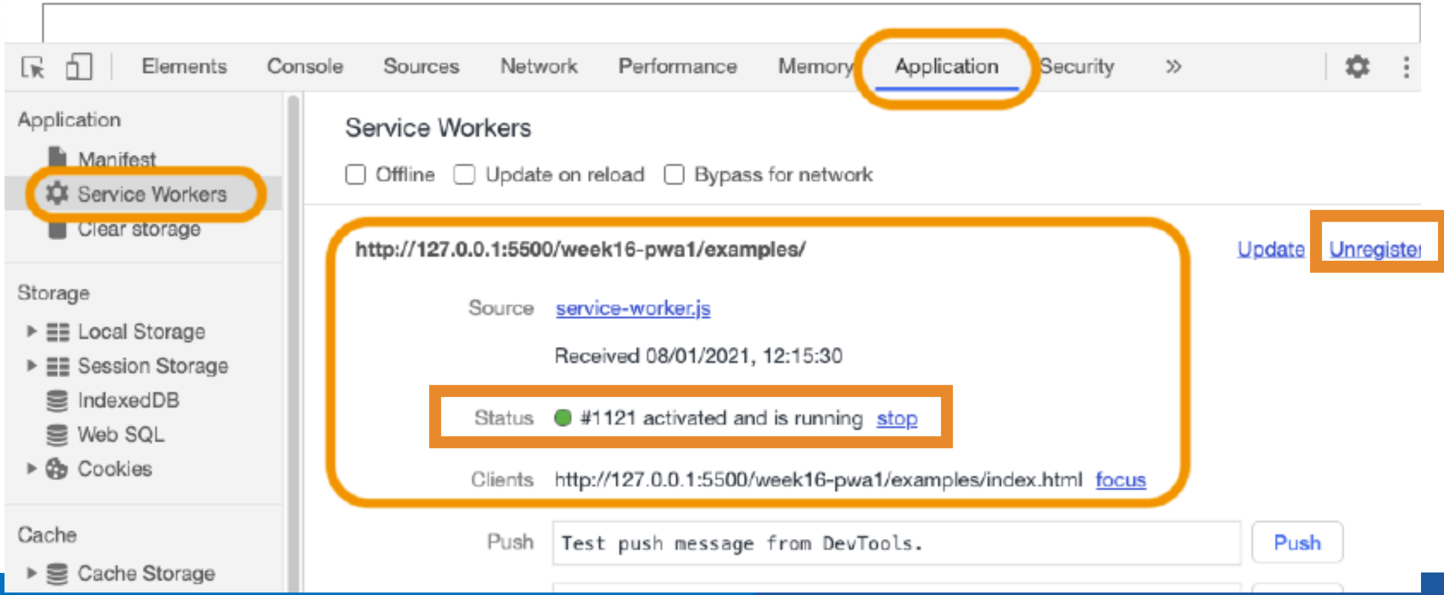
Yarn your cat can play with for a very **long** time!

Price: 2.99

Available stock: 7

Add to cart Buy now!

★★★☆☆



Service Workers

☐ Offline ☐ Update on reload ☐ Bypass for network

http://127.0.0.1:5500/week16-pwa1/examples/ Update Unregister

Source service-worker.js

Received 08/01/2021, 12:15:30

Status ● #1121 activated and is running stop

Clients http://127.0.0.1:5500/week16-pwa1/examples/index.html focus

Push `Test push message from DevTools.` Push

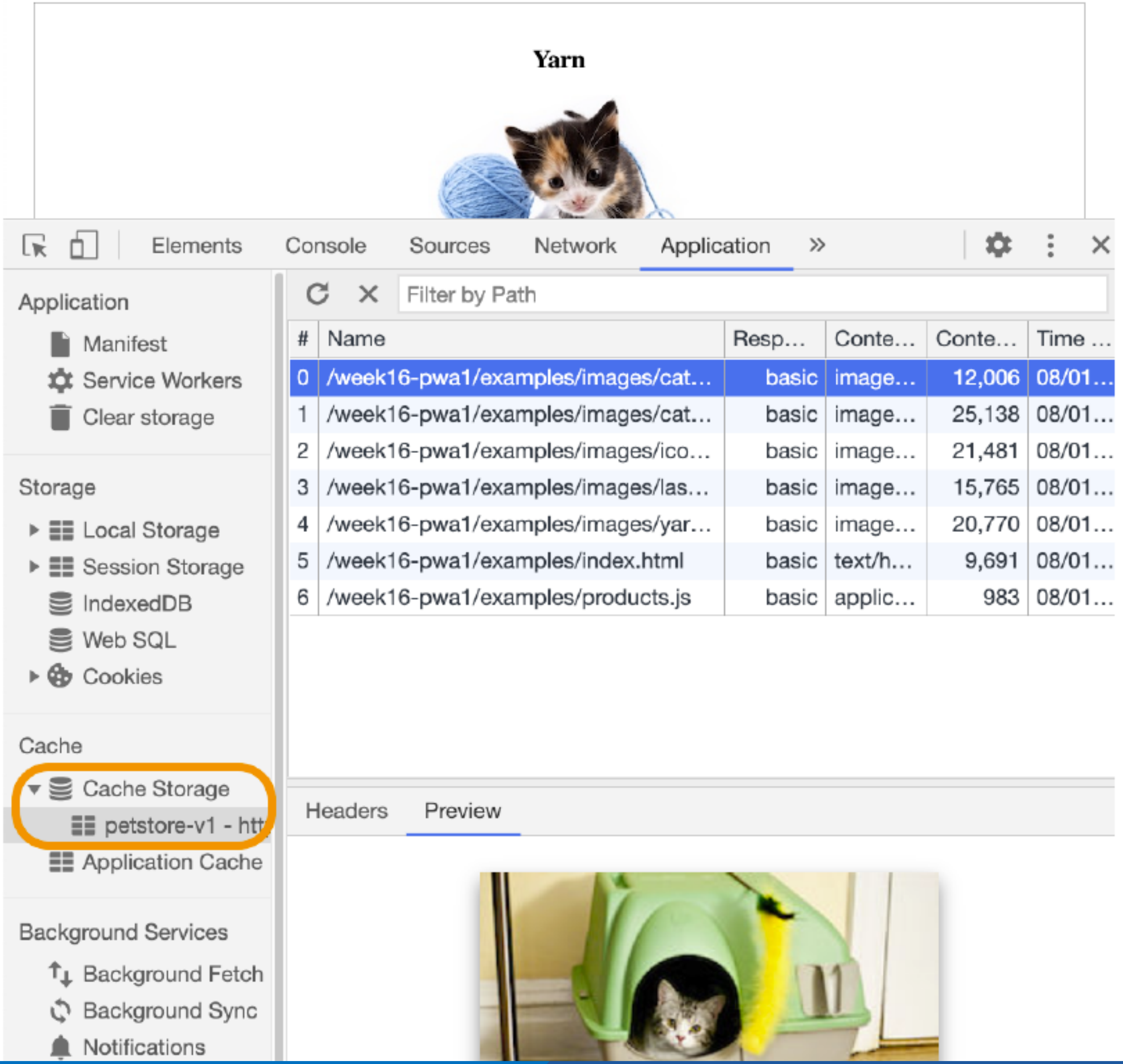
The Cached Files

Let’s see the file Cached by the the ServiceWorker:

- If you select 'Cache Storage' from the left pane, there should a cache named `petstore-v1`

- Clicking on `petstore-v1` will show the list of files that have been added to the cache in the top right pane

- Clicking on any of the file will show a preview in the bottom right pane

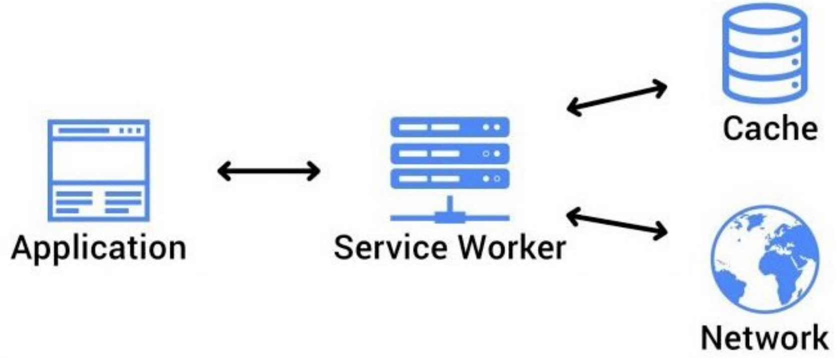


#	Name	Resp...	Conte...	Conte...	Time ...
0	/week16-pwa1/examples/images/cat...	basic	image...	12,006	08/01...
1	/week16-pwa1/examples/images/cat...	basic	image...	25,138	08/01...
2	/week16-pwa1/examples/images/ico...	basic	image...	21,481	08/01...
3	/week16-pwa1/examples/images/las...	basic	image...	15,765	08/01...
4	/week16-pwa1/examples/images/yar...	basic	image...	20,770	08/01...
5	/week16-pwa1/examples/index.html	basic	text/h...	9,691	08/01...
6	/week16-pwa1/examples/products.js	basic	applic...	983	08/01...

PWA, Service Workers and Dynamic Caching

Using the Cached Files in a Static Way

- **[Static Caching]**: Using the Cached Files in a Static Way
- Now that the files are cached, we can **use the local files when starting the app** rather than retrieving them from the server
  - This is **faster** and also allow the app **to work offline**
- The ways to get this to work is to **'intercept'** any `fetch` **request**
  - When the front end sends any `fetch` request to get data from the server
  - The service worker will redirect the request to the cache and return the file there
- This is achieved by listening to the fetch event, we respond to the fetch event with a function that tries to find the resource in the cache and return the response if it is there.
- The `FetchEvent.respondWith` method intercepts all `fetch` request
- functions as a proxy server between the app and the network.

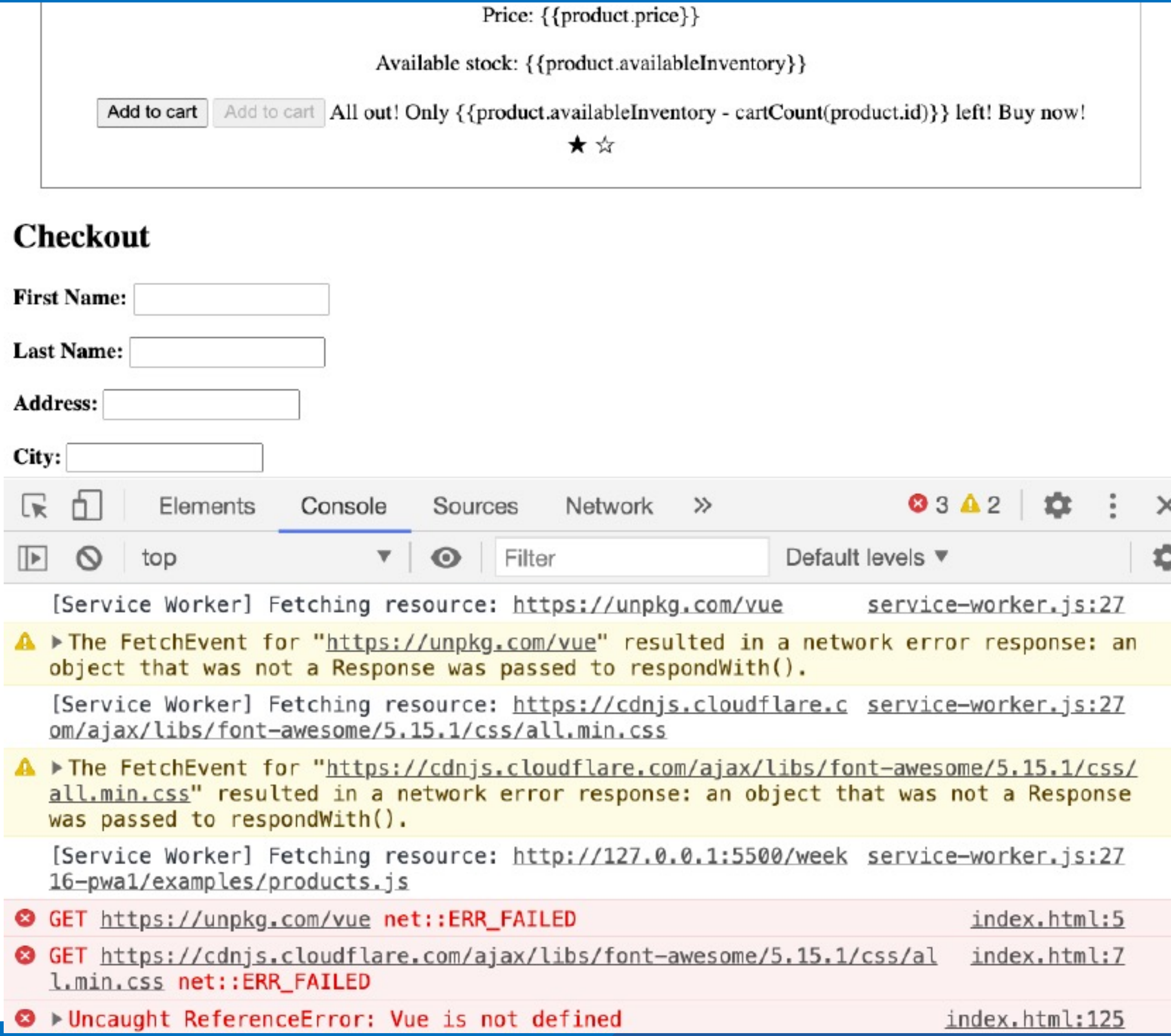


```
self.addEventListener('fetch', function (e) {
   e.respondWith(
      // check if the cache has the file
      caches.match(e.request).then(function (r) {
         console.log('[Service Worker] Fetching resource: ‘ + e.request.url);
         // 'r' is the matching file if it exists in the cache
         return r
      })
   );
});
```

Problem when Just Enabling Static Caching

Problem when Just Enabling Static Caching:

- there are still a few errors if you **refresh the page**
- when refreshed, all the files are loaded from the cache
  - because **we intercept every fetch request**
- While **html and images are there**
  - Third-party files are missing
  - Such as the Vue.js and css for FontAwesome



Price: {{product.price}}

Available stock: {{product.availableInventory}}

Add to cart Add to cart All out! Only {{product.availableInventory - cartCount(product.id)}} left! Buy now!

★ ☆

Checkout

First Name:

Last Name:

Address:

City:

```
[Service Worker] Fetching resource: https://unpkg.com/vue        service-worker.js:27
▶The FetchEvent for "https://unpkg.com/vue" resulted in a network error response: an
object that was not a Response was passed to respondWith().
[Service Worker] Fetching resource: https://cdnjs.cloudflare.c  service-worker.js:27
om/ajax/libs/font-awesome/5.15.1/css/all.min.css
▶The FetchEvent for "https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/
all.min.css" resulted in a network error response: an object that was not a Response
was passed to respondWith().
[Service Worker] Fetching resource: http://127.0.0.1:5500/week  service-worker.js:27
16-pwa1/examples/products.js
GET https://unpkg.com/vue net::ERR_FAILED                                 index.html:5
GET https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/al    index.html:7
l.min.css net::ERR_FAILED
▶Uncaught ReferenceError: Vue is not defined                            index.html:125
```

Dynamic Caching

Dynamic Caching: Caching New Files

```
self.addEventListener("fetch", function(e) {
   e.respondWith(
      caches.match(e.request).then(function (cachedFile) {
         //download the file if it is not in the cache
         if (cachedFile) {
            console.log("[Service Worker] Resource fetched from the cache for: " + e.request.url);
            return cachedFile;
         } else {
            return fetch(e.request).then(function (response) {
               return caches.open(cacheName).then(function (cache) {
                  //add the new file to the cache
                  cache.put(e.request, response.clone());

                  console.log("[Service Worker] Resource fetched and saved in the cache for: " +
e.request.url);
                  return response;
               });
            });
         }
      })
   );
});
```

Dynamic Caching: Code Explained

Dynamic Caching: Caching New Files

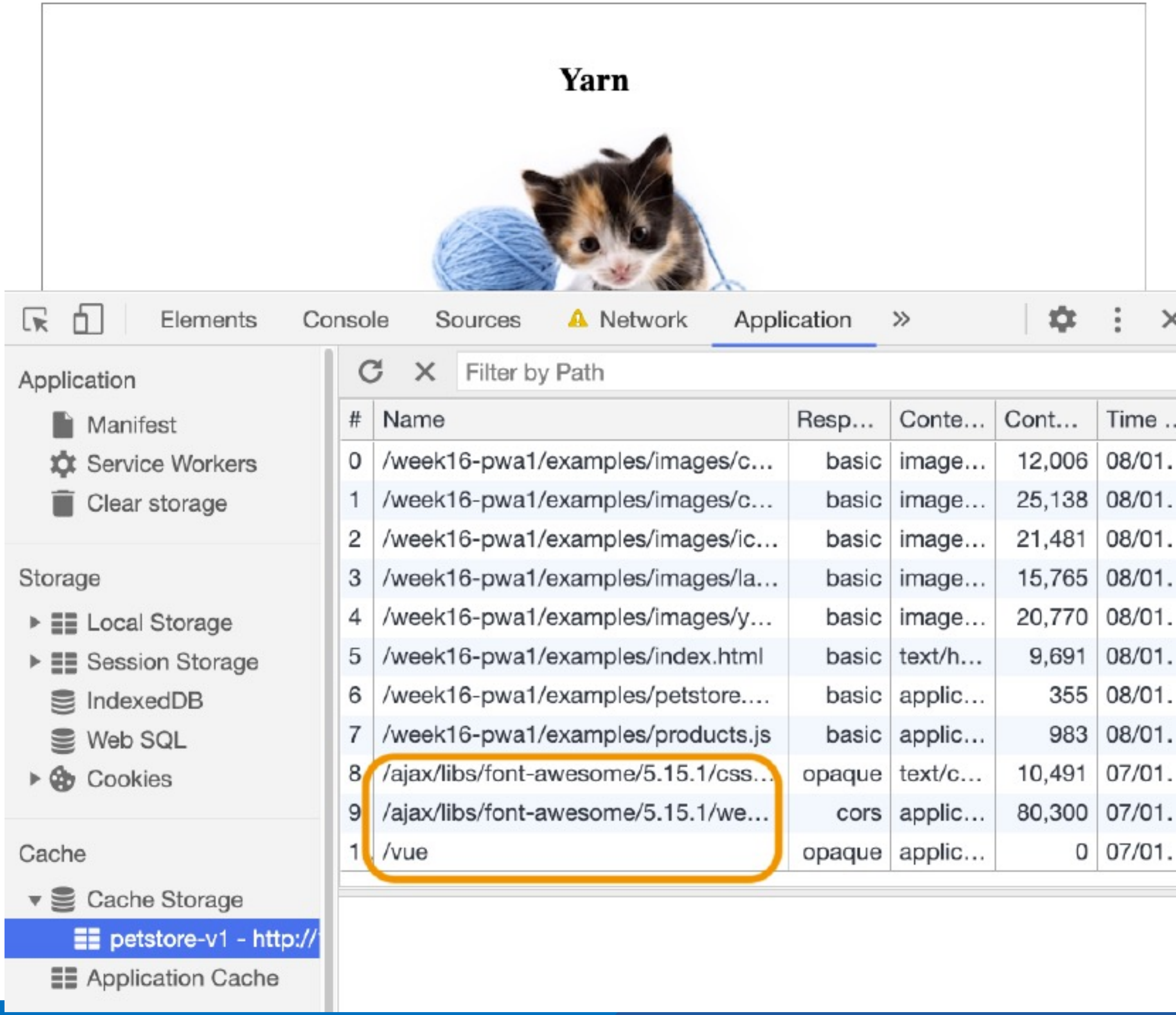
- While it is possible to add all the missing files to the caching file list
- The alternative is to download missing files dynamically and add them to cache

- `fetch(e.request)` : if the file is not in the cache, we use another fetch request to download it,
- `caches.open(cacheName).then(...)` : then store the response in the cache so it will be available there next time it is requested.

- This allows us to respond to every single request with any response we want:
  - **The service worker have full control of the response (and can be potentially used for malicious purpose)**

Dynamic Caching: Result

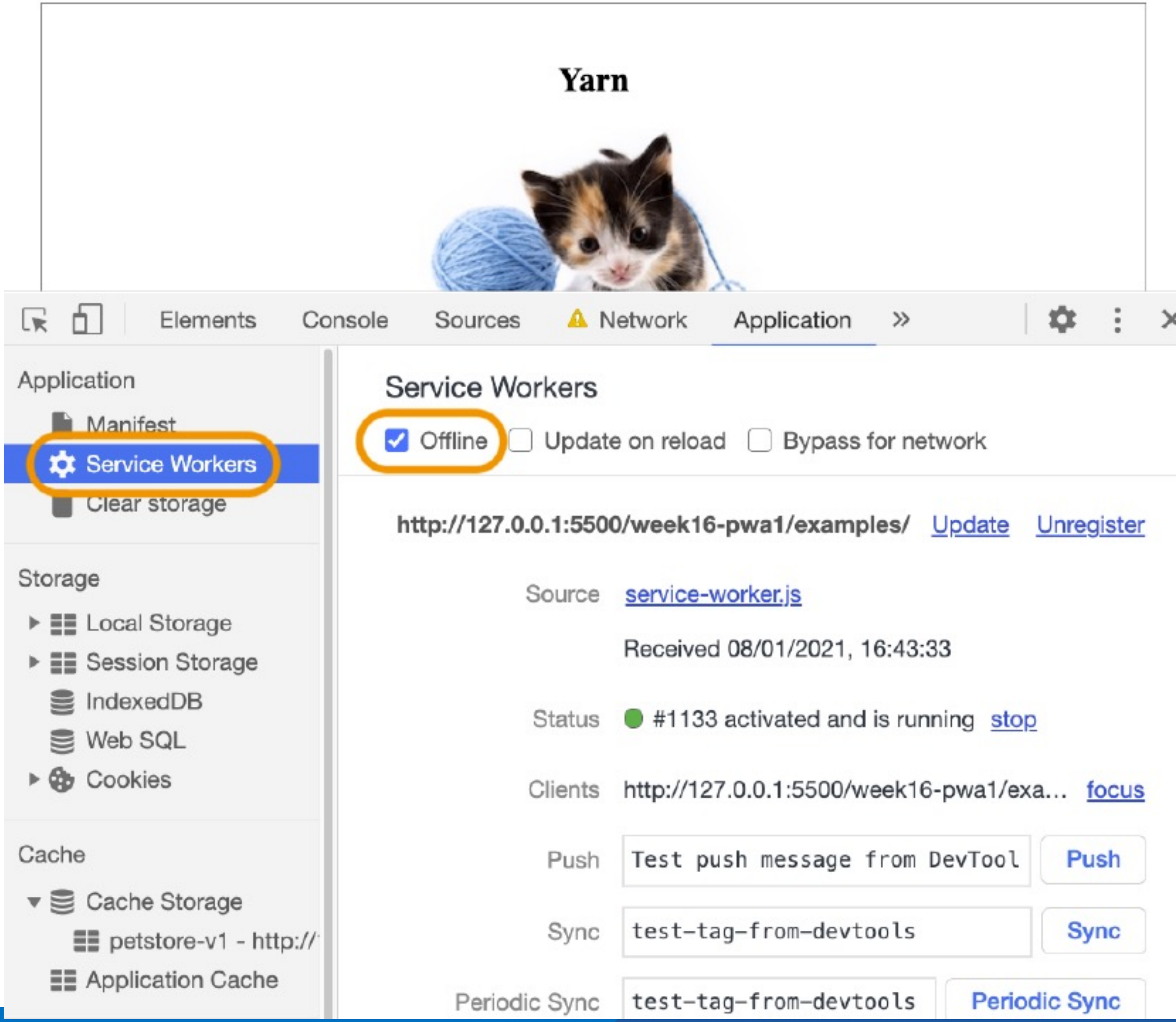
- **Now the missing files are added to the cache when the page reloaded:**
    - the files for FontAwesome and Vue are now added to the cache



#	Name	Resp...	Conte...	Cont...	Time ...
0	/week16-pwa1/examples/images/c...	basic	image...	12,006	08/01.
1	/week16-pwa1/examples/images/c...	basic	image...	25,138	08/01.
2	/week16-pwa1/examples/images/ic...	basic	image...	21,481	08/01.
3	/week16-pwa1/examples/images/la...	basic	image...	15,765	08/01.
4	/week16-pwa1/examples/images/y...	basic	image...	20,770	08/01.
5	/week16-pwa1/examples/index.html	basic	text/h...	9,691	08/01.
6	/week16-pwa1/examples/petstore....	basic	applic...	355	08/01.
7	/week16-pwa1/examples/products.js	basic	applic...	983	08/01.
8	/ajax/libs/font-awesome/5.15.1/css...	opaque	text/c...	10,491	07/01.
9	/ajax/libs/font-awesome/5.15.1/we...	cors	applic...	80,300	07/01.
1	/vue	opaque	applic...	0	07/01.

Testing the App Offline

- **You can also test this in the 'Service Workers' pane with the 'Offline' Option:**
  - this disconnects the network
  - and the app should still work fine after refresh



Yarn

Service Workers

☑ Offline □ Update on reload □ Bypass for network

http://127.0.0.1:5500/week16-pwa1/examples/ Update Unregister

Source service-worker.js

Received 08/01/2021, 16:43:33

Status ● #1133 activated and is running stop

Clients http://127.0.0.1:5500/week16-pwa1/exa... focus

Push Test push message from DevTool Push

Sync test-tag-from-devtools Sync

Periodic Sync test-tag-from-devtools Periodic Sync

Installable PWA, Testing it Locally and as a Mobile App, and HTTPS

Requirements Met

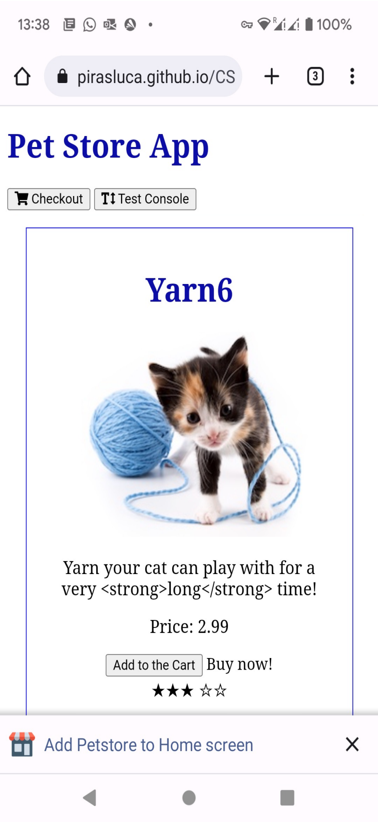
Now we met all the Requirements to our App an Installable PWA:

- ☑ **[Web Manifest]** a web manifest file, with the correct fields
- ☑ **[HTTPS]** the website needs to be on a **server with HTTPS connection**
    - we will use **GitHub Pages**, which provide HTTPS connection
- ☑ **[App Icon]** an icon (image file) to represent the app on the device
- ☑ **[Service Worker]** a registered service worker, to make the app work offline
    - requires some coding

Our PWA App Available on Your Phone

Now you can install our PWA App on your phone:

- Open Chrome,
- Indicate your GitHub Pages Url
- Install the App
- Open it from the icon in the home screen



13:38 100%

pirasluca.github.io/CS

Pet Store App

Checkout | Test Console

Yarn6

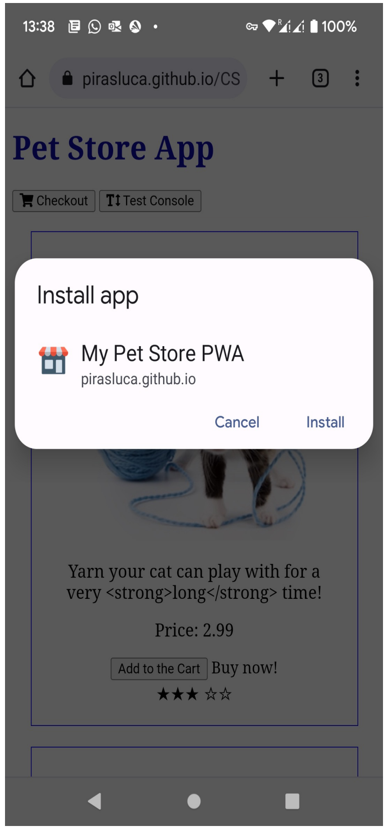
Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

Add to the Cart Buy now!

★★★ ☆☆

Add Petstore to Home screen ✕



13:38 100%

pirasluca.github.io/CS

Pet Store App

Checkout | Test Console

Install app

My Pet Store PWA

pirasluca.github.io

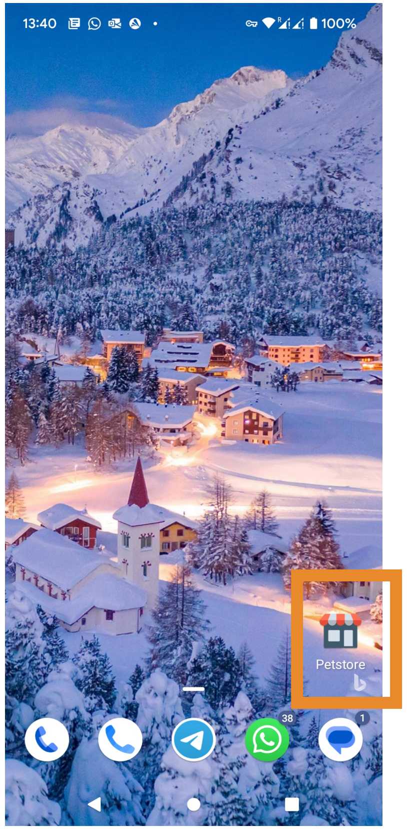
Cancel Install

Yarn your cat can play with for a very <strong>long</strong> time!

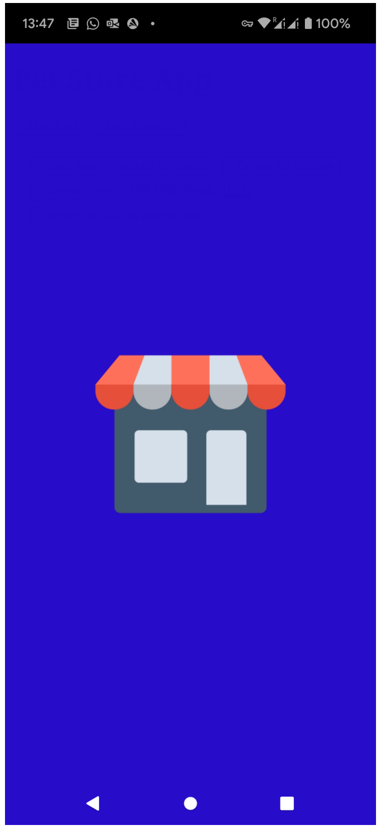
Price: 2.99

Add to the Cart Buy now!

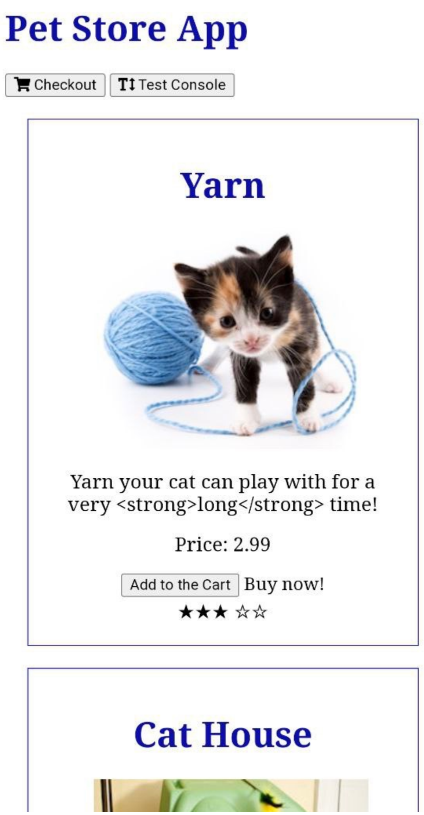
★★★ ☆☆



13:40 100%



13:47 100%



Pet Store App

Checkout | Test Console

Yarn

Yarn your cat can play with for a very <strong>long</strong> time!

Price: 2.99

Add to the Cart Buy now!

★★★ ☆☆

Cat House

Note, at the moment:

- Lesson.js is used for the app and it is not connected to AWS, because of issues with our Self-Signed Testing Certificate for HTTPS
- Later we will fix also this ☺

Suggestions for Reading

Reading

- MDN: Introduction to Progressive Web Apps: link

Questions?